SD Theater Ticketing System

Software Requirements Specification

Version 1

02/07/2024

Group #11
Alberto Escalante, Alex Thompson, Isaac Reveles

Prepared for
CS 250 - Introduction to Software Systems
Instructor: Gus Hanna, Ph.D.
Fall 2023

Revision History

Date	Description	Author	Comments
<date>	<Version 1>	<Your Name>	<First Revision>

Document Approval

The following Software Requirements Specification has been accepted and approved by the following:

Signature	Printed Name	Title	Date
	<Your Name>	Software Eng.	
	Dr. Gus Hanna	Instructor, CS 250	

Table of Contents

Revision History ii

Document Approval ii

1. Introduction 1

1.1 Purpose 1
1.2 Scope 1
1.3 Definitions, Acronyms, and Abbreviations 1
1.4 References 1
1.5 Overview 1

2. General Description 2

2.1 Product Perspective 2
2.2 Product Functions 2
2.3 User Characteristics 2
2.4 General Constraints 2
2.5 Assumptions and Dependencies 2

3. Specific Requirements 2

3.1 External Interface Requirements 3
3.1.1 User Interfaces 3
3.1.2 Hardware Interfaces 3
3.1.3 Software Interfaces 3
3.1.4 Communications Interfaces 3
3.2 Functional Requirements 3
3.2.1 <Functional Requirement or Feature #1> 3
3.2.2 <Functional Requirement or Feature #2> 3
3.3 Use Cases 3
3.3.1 Use Case #1 3
3.3.2 Use Case #2 3
3.4 Classes / Objects 3
3.4.1 <Class / Object #1> 3
3.4.2 <Class / Object #2> 3
3.5 Non-Functional Requirements 4
3.5.1 Performance 4
3.5.2 Reliability 4
3.5.3 Availability 4
3.5.4 Security 4
3.5.5 Maintainability 4
3.5.6 Portability 4
3.6 Inverse Requirements 4
3.7 Design Constraints 4
3.8 Logical Database Requirements 4
3.9 Other Requirements 4

4. Analysis Models 4

4.1 Sequence Diagrams 5
4.3 Data Flow Diagrams (DFD) 5
4.2 State-Transition Diagrams (STD) 5

5. Change Management Process 5

A. Appendices 5

A.1 Appendix 1 5
A.2 Appendix 2 5

SD Theater Ticketing System

1. Introduction

1.1 Purpose

The Software Requirement Specification document is to describe the layout of the SD Theater Ticketing System and the requirements needed in order to bring this product to life. This document explains the purpose and requirements of how this system will function to the software developers, stakeholders, and the administrators of this system.

1.2 Scope

The SD Theater Ticketing System will be a website that can be accessed through any search engine such as Google, Bing, etc., that allows customers to purchase movie tickets. The goal of this system is to provide customers a functionable platform where they can search movies, check different SD Theater locations in San Diego with their showtimes, and in the end purchase tickets. This system is to also be used by SD Theater employees and administrators. The employees will be able to update the list of movies and change the price of tickets as well as the showtimes. The administrators will have ultimate access to the system, but will mainly focus on gathering metrics such as how many tickets were sold online, how much revenue was made, and processing customer accounts along with any financial transactions.

1.3 Definitions, Acronyms, and Abbreviations

SD: San Diego
JOSN: JavaScript Object Notation
XML: Extensible Markup Language
SRS: Software Requirement Specification

1.4 References

No references are being used for this Software Requirement Specification document.

1.5 Overview

The rest of this SRS document contains the perspective of how this product will look like, the functions of this product, and any constraints that we can take into consideration. This document will go step-by-step into describing the layout by discussing the product's characteristics, attributes, functions, requirements, and the most possible use cases.

2. General Description

2.1 Product Perspective

2.2 Product Functions

The functions that the SD Theater Ticketing System will perform will be asking the customer to create an SD Theater Account or to sign in as a guest. The system will show a list of movies, showtimes, and seats that are available. There will be a payment process that will ask the customer to provide their credit/debit card information, apple pay, or paypal to authorize the purchase. An email confirmation will be sent to the customer's email address. The system will allow administrators to process customer account information such as transactions and to secure private customer information. It will also generate current metrics at any time and automatic reports of monthly/yearly performances such as revenue made, tickets sold, and activity of the website.

2.3 User Characteristics

The **customers** will be the individuals who are clicking on the website to browse movies and to purchase tickets. The SD Theater **employees** will be in charge of updating which movies to be shown, the showtimes of each movie and location, the seat availability, and attending customers if they have questions. The **administrators** will handle payment transactions, account information, and gather performance data reports.

2.4 General Constraints

1.) SD Theater Ticketing Systems requires a reliable internet connection in order to be accessed.
2.) The system will have a limit of 30 tickets per purchase, purchasing 31 tickets or more will require separate transactions.
3.) The system will save all customer information data and transactions within the SD Theater database

2.5 Assumptions and Dependencies

Assumptions:

- Customers will be able to access and navigate through the system's website as long as they have access to an internet connection.
- The SD Theater database will have updated information on all the movies.
- Customers are able to have a smooth checkout process.

Dependencies:

- The overall system depends on maintenance and updates to avoid slow interface functionality and processing
- The customer can have a smooth checkout, but the system won't authorize the purchase if there is insufficient funds. The system depends on accurate transaction processing and detection.
- The movies on the system's website depend on the updated movie information within the SD Theater database.

3. Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interfaces

The user interface will consist of the customer interacting with the system's website and making sure that it functions properly.

3.1.2 Hardware Interfaces

The SD Theater Ticketing System can be accessed through a computer, laptop or smartphone. The use of a computer or laptop is recommended for a better experience.

3.1.3 Software Interfaces

The system will have a URL since it is a website, therefore, it will need to be accessed through a search engine such as Google Chrome, Bing, FireFox, etc.

3.1.4 Communications Interfaces

A wireless internet connection will be the main source of the communication service that will allow the service to connect with the individual's hardware and software.

3.2 Functional Requirements

3.2.1 Creating the Account

1.) The homepage of the system's website will begin by asking the customer to create an account by entering an email address and password, to continue as a guest, or to sign in if they already have an account.
2.) The system should have two links at the bottom of the homepage, an "Employee Login" and "Administrator Login" link. These options will require their employee ID numbers to ensure authorized access.
3.) Once the account is created, it should be stored within the SD Theater Ticketing System employee database.
4.) The system should provide the option of resetting their password within the homepage and a "Forgot Password?" link on the login/signup page.

3.2.2 Booking a movie

1.) The website's homepage should have a collection of movies that are new and most popular.
2.) There should be two tabs, one that says "genres" and another that says "all movies".
3.) The "genres" tab will have a selection of all the movie genres along with the appropriate movies that fall under each of them.
4.) The "all movies" tab will simply allow the user to browse all the movies that are currently playing.
5.) Once the customer clicks on the desired movie, the information for that movie should be displayed such as what date to attend, the locations that are showing that movie, and their showtimes, and the duration of the movie.

6.) The system should ask how many tickets they would like to purchase by asking how many children, adults, or seniors are attending. The price will vary for each group.
7.) After the location and showtime is selected, the system should then provide an animation of all the seats in the theater room, identifying which ones are available and unavailable.
8.) The system should have a tab at the bottom right corner that says “checkout”.
9.) The checkout page should provide the customer different payment methods such as credit/debit card, apple pay, and paypal. It should also show the quantity of tickets being purchased and the price of each.
10.) There should be a tab that says “submit” that will process and authorize the purchase.
11.) After the payment is authorized, the system should send an email confirmation to the customer’s email address with the reservation number, the date and time of the screening, the seat number(s), name of the movie, and the total amount paid.
12.) The system should have the option for a customer to cancel their ticket reservation by having a tab on their profile that says “cancel reservation”.

3.2.3 Movie Updates by Employee

1.) After the employee signs in with their email and password, the system should redirect them to a different page to enter their employee ID number.
2.) On the Employee Homepage, there should be tabs with unique options such as “add/remove movies”, “add screen time”, and “update ticket price”.
3.) The “add/remove movie” tab should allow the employee to either add a new movie or remove a movie at their specific theater location.
4.) The “add screen time” tab should provide the employee the action to add different screen times for new or current movies.
5.) The “update ticket price” tab should allow the employee to update the prices of tickets, varying for children, adults, and seniors.

3.2.4 Administrative Control

1.) The system should provide a space for administrators to enter their administrator ID.
2.) After successful login, the administrator will be taken to their homepage.
3.) The system should provide administrators with ultimate access, meaning they also have the same options as the employees.
4.) There should be unique tabs such as “generate metric report” and “security hub”.

3.3 Use Cases

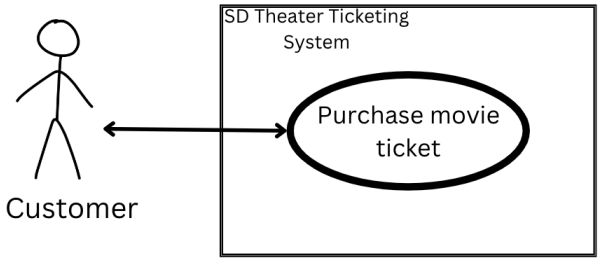
3.3.1 Use Case #1: Movie Ticket Purchase

Use case: Customer purchases a movie ticket
Actor: Customer
Procedure:

1.) Customer logs in, creates an account, or continues as a guest.
2.) Customer browses movies by genre or by all movies.
3.) Customer selects a movie and specific SD Theater location.
4.) Customer will select available screen time and amount of tickets by entering the number of children, adults, or seniors attending.

5.) Customer chooses available seats that correspond to the quantity of tickets.
6.) Customer clicks on the “checkout” tab and is taken to the checkout page.
7.) Customer pays.
8.) System will determine if the purchase was approved or unapproved. If approved, then an email confirmation with the digital ticket(s) is sent to the customer’s email address.

Diagram:

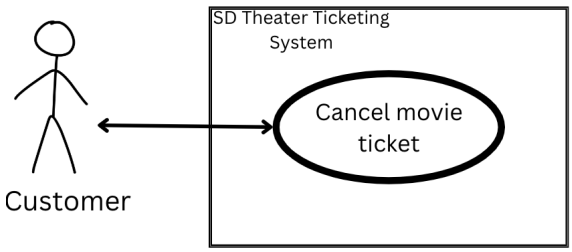


3.3.2 Use Case #2 Ticket Cancellation/Modification

Use case: Customer cancels their movie reservation
Actor: Customer
Procedure:

1.) Customer logs into their account.
2.) Customer clicks on their profile tab and selects “tickets”.
3.) Customer will see their upcoming event(s).
4.) Customer will click on the event they want to cancel.
5.) Customer cancels their event.
6.) Customer will receive the necessary reimbursement and an email to confirm the cancellation of their ticket reservation(s).

Diagram:



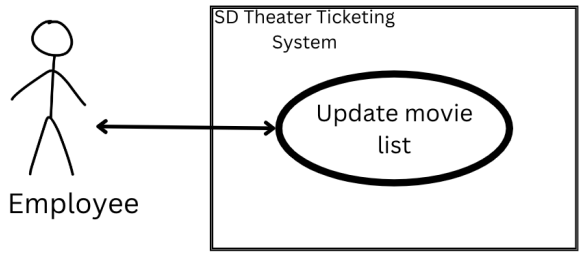
3.3.3 Use Case #3: Movie List Update

Use case: SD Theater employee updates the list of movies to be shown in theater.
Actor: SD Theater employee
Procedure:

1.) Employee logins into their account along with their employee ID.

2.) Employee is taken to their homepage.
3.) Employee clicks on the “add/remove movies” tab.
4.) The system will connect to SD Theater’s movie database.
5.) The system will update its movie list based on the up-to-date data.
6.) Employee will receive a confirmation of the movie list update.

Diagram:

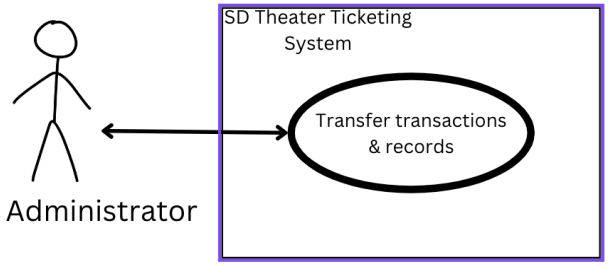


3.3.4 Use Case #4 Account Security & Management

Use case: Administrator transfers financial transactions and records into database.
Actor: SD Theater administrator
Procedure:

1.) Administrator logins into their account with their administrator ID.
2.) Administrator is taken to their homepage.
3.) Administrator clicks on “security hub”.
4.) Administrator then transfers data from the financials transactions and records into SD Theater’s database.
5.) SD Theater’s database gets updated.
6.) Administrator receives confirmation of the data transfer.

Diagram:



3.4 Classes / Objects

Attributes:

3.4.1 Ticket

- Seat number
- Booking status
- Price

3.4.2 Movie Information

- Title
- Duration
- Genre
- Release date

3.4.3 Payment Information

- Payment method
- Transaction ID
- Payment status
- Total amount

3.4.4 Showtime Information

- Date
- Time
- Theater number
- Available seats

Functions:

3.4.5 User Registration

- Allows the customer to create a personal account for SD Theater.

3.4.6 Movie Search

- Allows the customer to search movies by genre or simply view all movies.

3.4.7 Seat Selection

- View a layout of the theater seating and select preferred seats.

3.4.8 Ticket Booking

- Purchase ticket(s) for specific movie and showtime.

3.4.9 Booking Confirmation

- Receive an email confirmation of the ticket booking that contains the digital movie tickets and transaction ID.

3.4.10 Ticket Cancellation

- Allows the customer to cancel their ticket reservation and will then follow up with a reimbursement.

3.5 Non-Functional Requirements

3.5.1 Performance

The system's website should be able to respond fast when clicking on tabs or refreshing the page.

- The system should only accept a maximum page load time of 3 seconds and we aim for an average of 1.5 seconds.
- Resolve any errors within 24 hours of detection.

3.5.2 Reliability

The system should be accessed at any time, excluding maintenance time frames, and should be responsive.

3.5.3 Availability

- Target system availability should be 99.9% with an allowable downtime of 32.7 minutes per month.
- Planned maintenance time frame of 2 hours per month, notified at least 48 hours in advance.
- Unplanned maintenance time frames due to unexpected errors should be minimal as possible, 2 to 3 hours.

3.5.4 Security

- Password policy will require users to create a strong password that is made up of letters and numbers, but no special characters are needed.
- All sensitive data such as financial transactions and records are to be encrypted as they are being stored in the database.
- Ensure that users who are customers have the least necessary permissions.
- Ensure that employees and administrators of SD Theater enter their unique ID number to access their permissions.

3.5.5 Maintainability

If small changes are to be made, the system should continue to be up and running. Scheduled and unscheduled maintenance will require the system to be down.

3.5.6 Portability

The system should be able to be accessed and responsive correctly from any device that has a web browser/search engine.

3.6 Inverse Requirements

There are no inverse requirements for the SD Theater Ticketing System.

3.7 Design Constraints

- The interface for mobile devices may behave differently due to the smaller screen size and limiting process power.
- The system must correspond to SD Theater's security policies and protocols.

3.8 Logical Database Requirements

A database will be necessary to store and manage data for SD Theater.

Data Format: Definitions of some different types of data within the database.

- Date & Time formats

- Movie
- Currency formats
- Text formats
- Images
- JSON, XML files

Storage Capabilities: Definitions to accommodate the expected volume of data.
- Scalability
- Redundancy & Fault Tolerant
- Data Processing & Retrieval

Data Retention: Defines policies for retaining and archiving data.
- Retention periods for different types of data
- Archiving mechanisms for storing historical data offline or in secondary storage
- Deletion of obsolete or redundant data in order to free up storage

Data Integrity: Defines mechanisms for ensuring integrity and consistency within the database.
- Primary key constraints to enforce uniqueness of records
- Foreign key constraints to maintain integrity between tables

Data Security: Defines measures that will protect data from unauthorized access and tampering.
- Role-based access control to restrict access to sensitive data based on user roles and permissions
- Encryption of sensitive data when still and during transfer to prevent unauthorized interception or tampering
- Auditing and logging mechanisms to track and monitor access to data and detect suspicious activities

Backup & Recovery: Defines strategies for backing up and restoring data to ensure continuity in business operating and minimize data loss/corruption.
- Regular backups of the database to onsite or offsite storage
- Incremental backups to capture changes since the last full backup
- Recovery procedures to restore data in the event of hardware failure, software errors, or natural disasters

Data Access & Reporting: Defines mechanisms for accessing and querying data within the database.
- SQL queries and stored procedures for retrieving and manipulating data
- Integration with analytics platforms for extracting insights from trends in data

3.9 Other Requirements

Catchall section for any additional requirements.

4. Analysis Models

5. Change Management Process

Identify and describe the process that will be used to update the SRS, as needed, when project scope or requirements change. Who can submit changes and by what means, and how will these changes be approved.

A. Appendices

Appendices may be used to provide additional (and hopefully helpful) information. If present, the SRS should explicitly state whether the information contained within an appendix is to be considered as a part of the SRS's overall set of requirements.

Example Appendices could include (initial) conceptual documents for the software project, marketing materials, minutes of meetings with the customer(s), etc.

A.1 Appendix 1

A.2 Appendix 2

Software Design Specification

Alejandro Thompson, Alberto Escalante, Isaac Reveles

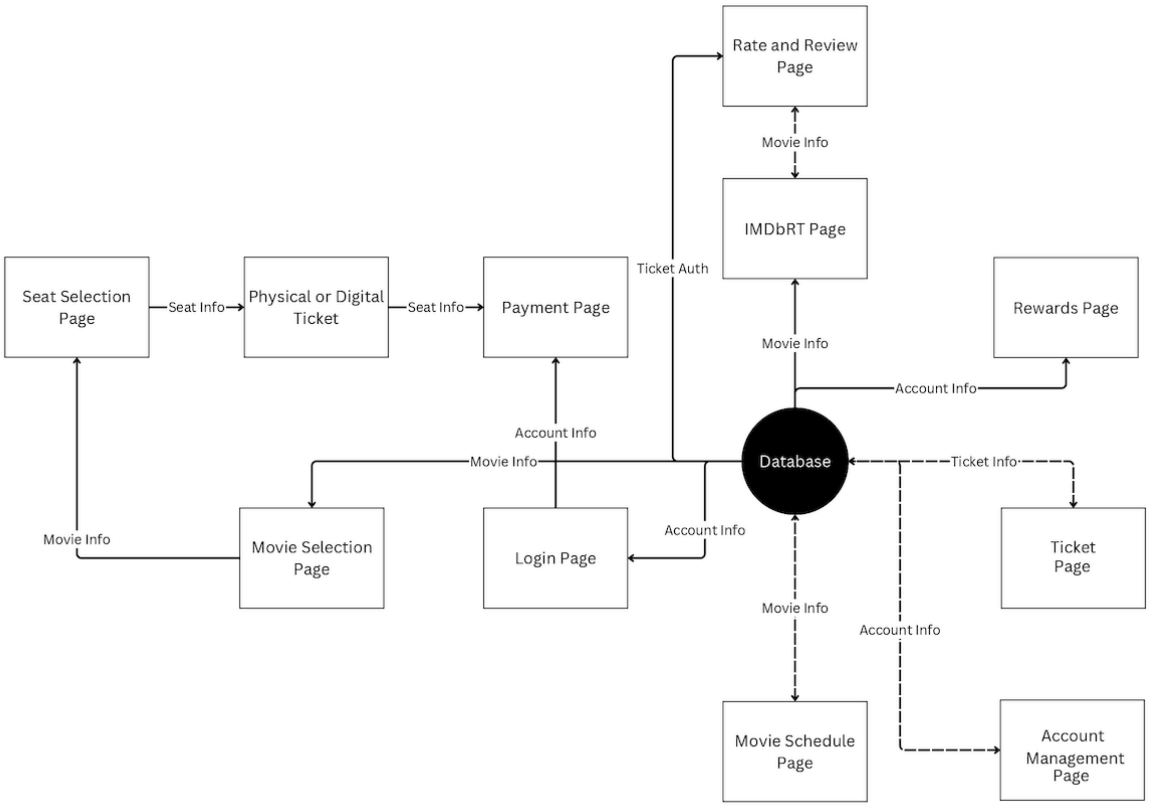
System Description

Brief overview of the system:
The SD Theater Ticketing System is an online system designed using an architectural diagram and a UML diagram. This Software Design Specification is intended to be a comprehensive guide that will showcase the detail that goes into the architecture, functionality and overall the structure of the system. This documentation is for the developers to see the correlation between each component of the ticketing system.

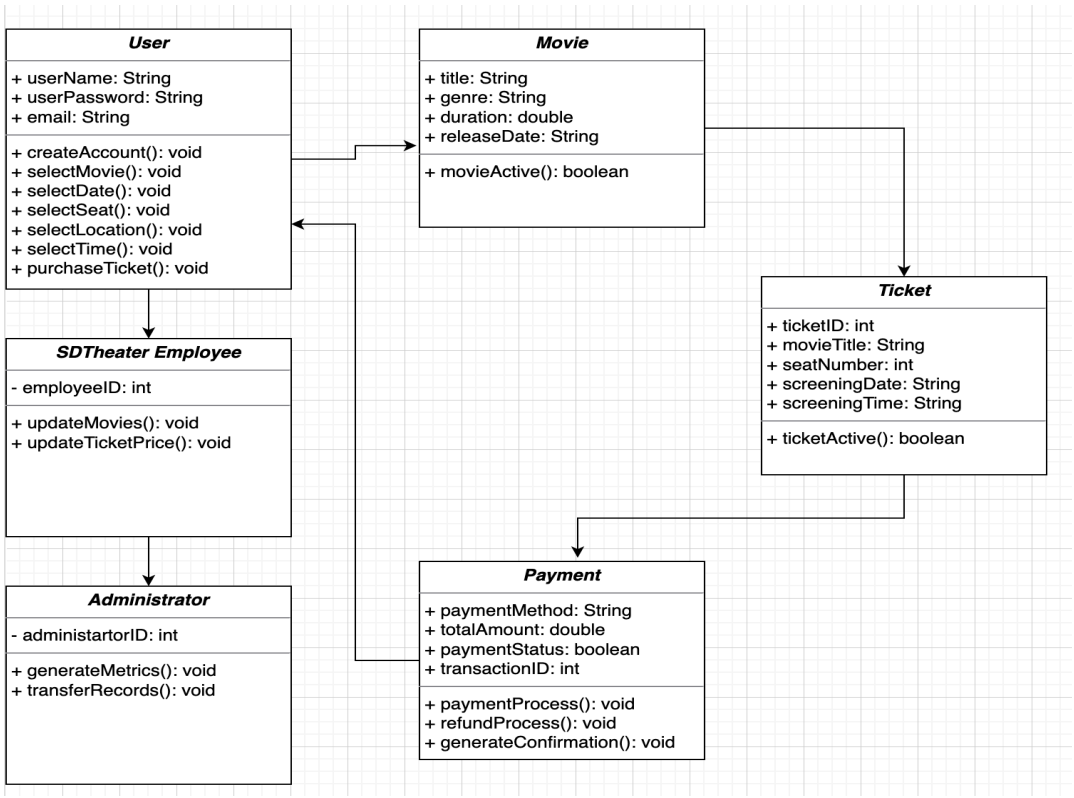
Software Architecture Overview

For the architecture of the system, it is based on the 3-layered data access approach; which are in this case Presentation Layer, Business Logic, and Database. The architecture below showcases the correlation between each layer.

Architecture Diagram:



UML Class Diagram:



Description of classes:

User: Represents users of the system, including customers and administrators. This class may have attributes such as ***Username, Password, Email, Role, etc.***

Admin: Represents an administrator of the system responsible for managing performances, venues, tickets, etc. This class may have attributes such as ***Username, Password, Permissions, etc.***

Movie: Represents the showing of the movie. This class may have attributes such as ***Title, Date, Time, Venue, Description of movie, etc.***

Employee: Represents the employees of the SD Theatre. This class may have attributes such as ***Employee ID, etc.***

Ticket: Represents a ticket purchased by a customer for a specific movie. This class may have attributes such as ***Ticket ID, Movie ID, Price, etc.***

Payment: Represents a payment transaction made by a customer to purchase tickets. This class may have attributes such as ***Payment ID, Amount, Payment Method, Payment Status, etc.***

Description of Attributes:

User:

1.) **userName**: User chooses their preferred name for their SD Theater account. Represented as a String.
2.) **userPassword**: User enters their password for verification. Represented as a String.
3.) **email**: User's email address where they will be able to receive notifications and confirmations in regards to their account. Represented as a String.

SDTheaterEmployee:

1.) **employeeID**: SD Theater employees have the same user controls, the only additional information presented is their unique employee ID which gives them staff access to the system. Represented as an int that is private.

Administrator:

1.) **administratorID**: Administrators have the same access as customers and theater employees, the only new information needed is their administrator ID in order to have administrator access. Represented as an int that is private.

Movie:

1.) **title**: The name of the movie. Represented as a String.
2.) **genre**: The classification of the movie. Represented as a String.
3.) **duration**: The time span of the movie. Represented as a double.

4.) **releaseDate**: The date the movie will be released to the public in theaters. Represented as a String.

Ticket:

1.) **ticketID**: The unique ID for every ticket. Represented as an int.
2.) **movieTitle**: The name of the movie associated with the ticket. Represented as a String.
3.) **seatNumber**: Unique number of the seat associated with the ticket. Represented as an int.
4.) **screeningDate**: The date of when the movie will be played. Represented as a String.
5.) **screeningTime**: The time at which the movie will be played. Represented as a String.

Payment:

1.) **paymentMethod**: The source of payment the customer will use to pay. Represented as a String.
2.) **totalAmount**: The total amount that is due at checkout. Represented as a double.
3.) **paymentStatus**: Determines if the payment was accepted. Represented as a boolean.
4.) **transactionID**: The unique ID of every transaction made. Represented as an int.

Description of Operations:

User:

1.) **createAccount(): void**: This operation allows the user to create an SD Theater account by entering their username, password, and email address.
2.) **selectMovie(): void**: Enables the user to select any movie.
3.) **selectDate(): void**: Enables the user to select a date associated with the movie.
4.) **selectSeat(): void**: Enables the user to select any seat(s) that is/are available.
5.) **selectLocation(): void**: Enables the user to select any of the SD Theater locations in San Diego.
6.) **selectTime(): void**: Enables the user to select from a list of time slots that are available.
7.) **purchaseTicket(): void**: Allows the user to purchase number of tickets; depending on how many children, adults, or seniors are attending, lastly is taken to checkout.

SDTheaterEmployee:

1.) **updateMovies(): void**: Enables access for SD Theater employees to update the movie listing.
2.) **updateTicketPrice(): void**: SD Theater employees are able to change the price of tickets.

Administrator:

1.) **generateMetrics(): void**: Creates metrics of movies and how they have performed within the theater and system.
2.) **transferRecords(): void**: Allows access for security hub to be accessed and for business transactions and records to be transferred into the database.

Movie:

1.) **movieActive(): boolean**: Gets the status of any movie. Returns true if movie is still playing in theaters. Returns false if movie is not in theaters anymore.

Ticket:

1.) **ticketActive(): boolean**: Returns true if user ticket is currently active within the system.

Payment:

1.) **paymentProcess():void**: Runs payment process operation.
2.) **refundProcess():void**: If user requires refund for tickets, this operation runs the refund process.
3.) **generateConfirmation():void**: Confirms payment of tickets.

Development Plan and Timeline

To make sure the development of the SDTheatre software goes smoothly, we will be following a structured timeline as follows:

• **Weeks 1-2:** Project Planning, Location Verifications and Resource Requirements

• **Weeks 3-4:** Project Design and Beginning of Software Development

• **Weeks 5-6:** User and Employee Registration, Account Info and Login System Integration

• **Weeks 7-8:** Implementation of Payment System and PearPay Integration

• **Weeks 9-10:** Location, Theater and Seat Implementation

• **Weeks 11-12:** Ticket System Implementation and IMDbRT Integration

• **Weeks 13-14:** Security Implementation and Alpha Testing

• **Weeks 15-16:** Bug Fixes and Final Testing

• **Week 17:** Documentation and Beta Testing

• **Week 18:** Release Software

Team member responsibilities:

The following tasks will be assigned to team members as follows:

• **Project Manager:** Oversees that all objectives in the timeline are completed within parameters and that team members are completing their tasks efficiently.

• **Business Analyst:** Gather information and scope out the best locations and theaters for SDTheater and whether it can be verified there. Also verifies if the product is ready for commercial use.

• **Engineering Manager:** In charge with working with other team members to ensure a safe and a working product. Responsible of analyzing any challenges in hardware and figuring out fixes for them.

• **Software Architect:** In charge of conducting the internal arrangement of the software and work with the Software Developers to perfect the product with optimal technical solutions.

• **Software Developers:** Responsible of implementing the internal arrangement and systems in the software by working with the Software Architect and in parallel with UI Designers and QA Engineers.

• **UI Designers:** In charge of the Project Design and UI system of the software. Makes sure the user experience is optimal, easy and convenient to use. Also work in parallel with Engineer and Software Developers.

• **QA Engineer:** In charge of quality assurance during testing, makes sure to point out any flaws in the system and automate the testing process for software quality.

• **Testers:** Required to test the software in the Alpha and Beta phase to see if it is ready for commercial use. Provides feedback before the release of the software.

Verification Test Plan

The Verification Test Plan is intended to look over the SD Theater Ticketing System in ensuring that the functions are being properly executed and that the design specifications are being met. For the following test cases, each one is designed to focus on different features of the system and verifying that they meet the desired functionality.

SD Theater Ticketing System

1. **Test Case: User_Functions_0 (Create Account)**
    - **Test Features:** Functionality of user registrations
    - **Test Vectors:** Username, password with a minimum length of 8 characters, and email address
    - **What's being Covered?:** This test ensures that a new account can be created successfully and stored into the records database.

2. **Test Case: User_Function_1 (Select Movie)**
    - **Test Features:** Functionality of selecting a movie
    - **Test Vectors:** Movie being a part of the active movie database.
    - **What's being Covered?:** This test ensures that the user is able to search for a movie and being able to click on it.

3. **Test Case: User_Functions_2 (Select Date)**
    - **Test Features:** Available dates for a movie.
    - **Test Vectors:** Range of available, accurate dates for a specific movie.
    - **What's being Covered?:** This test case verifies that the user can choose from a list of available dates to watch their selected movie.

4. **Test Case: User_Functions_3 (Select Seat)**
    - **Test Features:** Available seats for a movie.
    - **Test Vectors:** Theater location, date, movie, and time chosen by user.
    - **What's Being Covered?:** This test case makes sure that the user is given a layout of the theater room with its available seats to choose from, depending on the theater location they have chosen.

5. **Test Case: User_Functions_4 (Select Location)**
    - **Test Features:** Viewing a list of SD Theater locations where the movie is being played.
    - **Test Vectors:** Movie chosen by user.
    - **What's being Covered?:** This test case ensures that the user is offered a list of SD Theater locations where the movie is being played.

6. **Test Case: User_Functions_5 (Select Time)**
    - **Test Features:** Viewing a list of times that are available for a movie.
    - **Test Vectors:** Movie and SD Theater location selected by user.
    - **What's being Covered?:** This test case verifies that the user is able to choose from a list of times that show when the movie will be played.

7. **Test Case: User_Functions_6 (Purchase Ticket)**
    - **Test Features:** Securing of a movie ticket.
    - **Test Vectors:** Desired movie, location, date, time, and seat.
    - **What's being Covered?:** This test ensures that the user is able to purchase a ticket after having selected their movie booking details.

8. **Test Case: Employee_Functions_0 (Update Movies)**
    - **Test Features:** Update list of movies.
    - **Test Vectors:** Selected movies to be added or removed through database.
    - **What's being Covered?:** This test case ensures that the employees are able to update any movie details to the system.

9. **Test Case: Employee_Functions_1 (Update Ticket Price)**
    - **Test Features:** Updating the price of tickets
    - **Test Vectors:** Tickets for children, adults, and seniors.
    - **What's being Covered?:** This test case verifies that employees have access to changing the ticket price for children, adults, and seniors.

10. **Test Case: Administrator_Functions_0 (Generate Metrics)**
    - **Test Features:** Generating a report of metrics from the ticketing system.
    - **Test Vectors:** Numerical data; number of tickets sold, number of registered users.
    - **What's being Covered?:** This test case makes sure that administrators are able to generate a metrics report for a specific range of dates.

11. **Test Case: Administrator_Functions_1 (Transfer Records)**
    - **Test Features:** Transferring business transactions and records to database.
    - **Test Vectors:** Personal data from guests.
    - **What's being Covered?:** This test case verifies that administrators have access to transferring records and transactions to the database.

12. **Test Case: Movie_Functions_0 (Movie Active)**
    - **Test Features:** Verifying if movie is still active.
    - **Test Vectors:** Desired or selected movie.
    - **What's being Covered?:** This test case ensures that the movie is still available or not.

13. **Test Case: Ticket_Functions_0 (Ticket Active)**
    - **Test Features:** Verifying if movie ticket is still active.
    - **Test Vectors:** Movie ticket purchased by user.
    - **What's being Covered?:** This test case is to verify if a movie ticket is still active or not.

14. **Test Case: Payment_Functions_0 (Payment Process)**
    - **Test Features:** Authenticates payment from user.
    - **Test Vectors:** User's form of payment.
    - **What's being Covered?:** This test case verifies and authenticates the user's payment at checkout.

15. **Test Case: Payment_Functions_1 (Refund Process)**
    - **Test Features:** Functionality of refund process.
    - **Test Vectors:** User's email address and their form of payment.
    - **What's being Covered?:** This test case is to ensure that the user successfully receives a refund with the correct amount.

16. **Test Case: Payment_Fuctions_2 (Generate Confirmation)**
    - **Test Features:** Functionality of email confirmation.
    - **Test Vectors:** User's email address and purchase of ticket.
    - **What's being Covered?:** This test case verifies that the user receives an email confirmation with the details of their ticket purchase.

Data Management Strategy

The SD Theater Ticketing System is made up of structured information with complex relationships which means strong consistency and accuracy are crucial components. Therefore, having a relational database such as SQL, would be the best fit. The data to be processed by the SD Theater Ticketing System will be stored in two separate SQL databases. One of the databases will be called 'User Database' and it will contain information that is related to the user. The second database will be called 'Business Database' and it will be made up of information regarding business activities of the SD Theater Ticketing System.

The total number of databases chosen is two, one of the user data and the other for the business data, in order to know which piece of data falls into the appropriate database. This approach ensures that the system can implement security requirements for the user data, provide efficiency in both databases, and overall create an optimal data management system.

The following information is a representation of how each database will function and what specific information is to be stored in each one:

User Database:

The ‘user database’ will contain personal information of the users of the ticketing system. The term ‘User’ is referred to customers, SD Theater employees, and the administrators.

1. **userName:** Chosen by the user when they are creating their account and to be made sure that it is unique. Stored as a String.
2. **userPassword**: Chosen by the user in order to authenticate their SD Theater account. Stored as a String.
3. **email:** Entered by the user that serves as a link for communication and security purposes. Stored as a String.
4. **employeeID:** Unique ID for every SD Theater employee which allows access to their portal on the ticketing system. Stored as an int.
5. **administratorID:** Unique ID for every admin; this gives them ultimate access over everything on the ticketing system and databases. Stored as an int.

Business Database:

The ‘business database’ will contain information and the attributes related to movies, booking tickets, and payment transactions. This database will be made up of three portions which are movie, ticket, and payment.

Movie:

1. **title:** Unique name of every movie that is playing or to be played will be stored. Stored as a String.
2. **genre:** Genre of every movie is stored as a String.
3. **duration:** Runtime of every movie is stored as a double.
4. **releaseDate:** The date of when a movie is to be released is stored as a String.

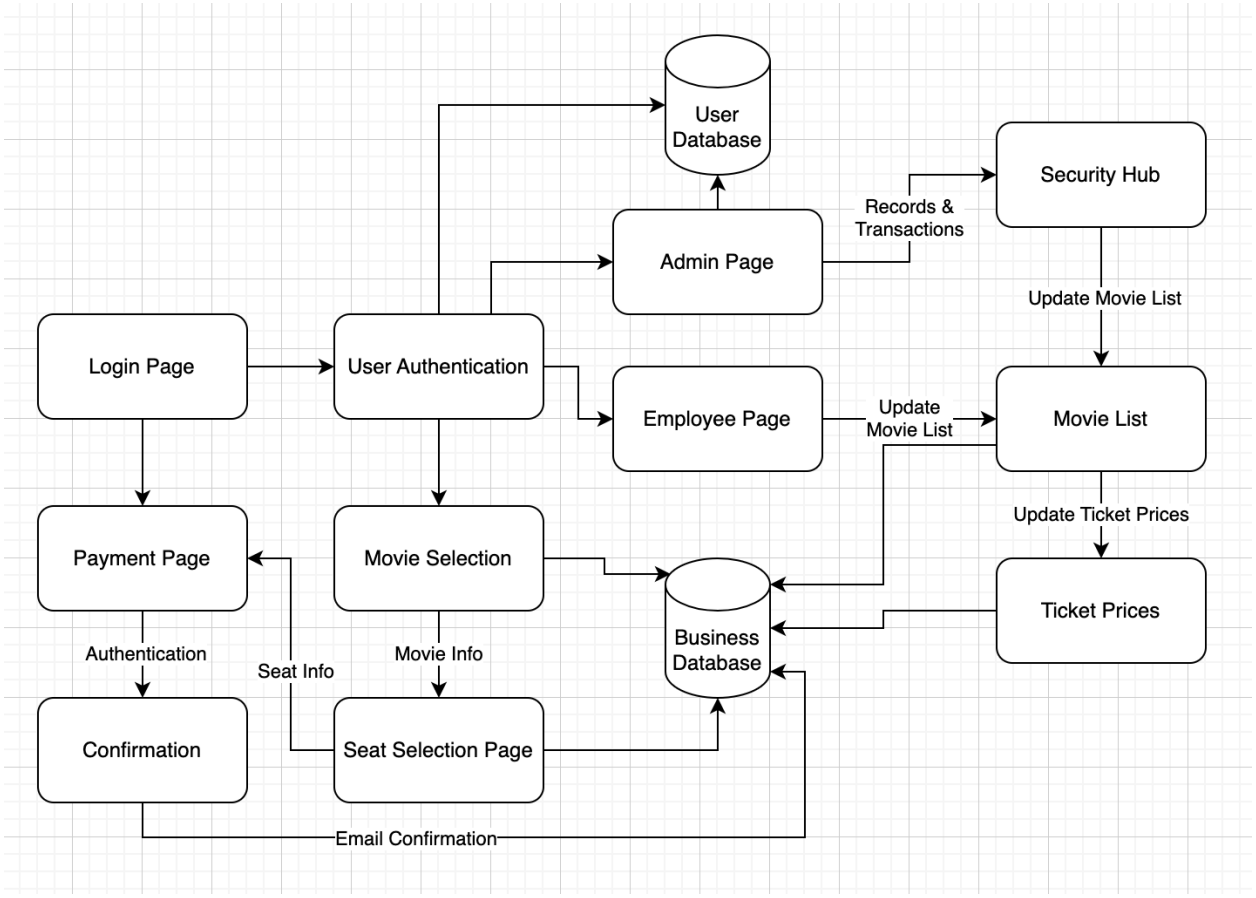
Ticket:

1. **ticketID:** Unique ID of every ticket booked is stored as an int.
2. **movieTitle:** Movie title that is shown and correlated to the ticket. Stored as a String.
3. **seatNumber:** The seat number chosen by the user is stored as an int.
4. **screeningDate:** The date chosen by the user of when the movie is to be played and when the ticket is valid. Stored as a String.
5. **screeningTime:** The time when the movie will begin to play. This time is correlated to the ‘screeningDate’ and ‘movieTitle’ attribute.

Payment:

1. **paymentMethod:** The form of payment that a user wishes to pay with. Stored as a String.
2. **totalAmount:** The total amount the user paid is part of a business transaction and can be considered as sensitive information. Stored as a double.
3. **paymentStatus:** Status that notifies if a payment has been approved is linked with the needed security requirements of the system. Stored as a boolean.
4. **transactionID:** Unique confirmation ID of every ticket purchased is to be stored as an int.

Software Architecture Diagram with Databases



Possible Alternatives

Splitting the data into separate databases or tables can have several advantages or disadvantages.

Some of these advantages include:

- Improved Data Organization
- Enhanced Data Integrity
- Better Performance
- Scalability
- Etc.

Some disadvantages:

- Increased Complexity
- Data Redundancy & Inconsistency
- Resource Overhead
- Complex Queries

Overall, while splitting the data into separate databases/tables, it gives us many advantages in terms of organization, performance, etc. While also having disadvantages in terms of complexity, redundancy, etc. Therefore, careful consideration and planning where necessary to determine the most suitable data grouping approach based on the specific requirements and constraints for SD Theater.